

Inverse-free Berlekamp–Massey–Sakata Algorithm and Small Decoders for Algebraic-Geometric Codes

Hajime Matsui, *Member, IEEE*, and Seiichi Mita, *Member, IEEE*

Abstract—This paper proposes a novel algorithm for finding error-locators of algebraic-geometric codes that can eliminate the division-calculations of finite fields from the Berlekamp–Massey–Sakata algorithm. This inverse-free algorithm provides full performance in correcting a certain class of errors, generic errors, which includes most errors, and can decode codes on algebraic curves without the determination of unknown syndromes. Moreover, we propose three different kinds of architectures that our algorithm can be applied to, and we represent the control operation of shift-registers and switches at each clock-timing with numerical simulations. We estimate the performance in comparison of the total running time and the numbers of multipliers and shift-registers in three architectures with those of the conventional ones for codes on algebraic curves.

Index Terms—codes on algebraic curves, syndrome decoding, Berlekamp–Massey–Sakata algorithm, Gröbner basis, linear feedback shift-register.

I. INTRODUCTION

ALGEBRAIC-GEOMETRIC (AG) codes, especially codes on algebraic curves, are comprehensive generalization of prevailing Reed–Solomon (RS) codes. They can be applied to various systems by choosing suitable algebraic curves without any extension to huge finite (Galois) fields. In fast decoding of such codes, Berlekamp–Massey–Sakata (BMS) algorithm [25] is often used for finding the location of errors, and the evaluation of error-values is done by using outputs of BMS algorithm with O’Sullivan’s formula [24].

RS codes have the features of high error-correcting capability and less complexity for the implementation of encoder and decoder. On the other hand, codes on algebraic curves have the issues related to the size of decoders as well as the operating speed of decoders. In particular, we notice that RS-code decoders need no inverse-calculator of the finite field (no finite-field inverter). The extended Euclidean algorithm [30] for RS codes has no divisions, and this enables us to operate compactly and quickly in calculating error-locator and error-evaluator polynomials. One inverse computation requires thirteen multiplications in practical $\text{GF}(2^8)$ and needs enormous circuit scale. Thus, it is strongly expected that the

fast inverse-free algorithm for AG codes will be established, since division operations are inevitable on the original BMS algorithm. In addition, the decoder that has small circuit-size, such as the conventional RS decoder, is considered necessary.

In this paper, we propose an inverse-free BMS algorithm, and give a whole proof of its adequacy. Moreover, we propose three kinds of small-sized architectures that generate error-locator polynomials for codes on algebraic curves. We then explain our architectures with model structures and numerical examples, and show the practical operation of proposed architectures in terms of the control flow of registers and switches at each clock-timing. The performance is estimated on the total running time and the numbers of multipliers and shift-registers for all architectures.

The divisions in the original BMS algorithm appear at the Berlekamp transform [1]

$$f_{N+1} := f_N - (d_N/\delta_N) g_N \quad (1)$$

at each N -loop in the algorithm, where f_N , g_N , and d_N are called minimal polynomial, auxiliary polynomial, and discrepancy at N , respectively, N runs over $0 \leq N \leq B$ for sufficiently large B , and δ_N is equal to a certain previous d_N . Then the inverse-free BMS algorithm consists of modified Berlekamp transforms of the form

$$f_{N+1} := e_N f_N - d_N g_N, \quad (2)$$

where e_N is equal to a certain previous d_N in this expression. Thus the denominator δ_N in (1) is converted into the multiplication of e_N in (2). This version of inverse-free BMS algorithm can be proved in the comparable line of the original algorithm. However, there is a significant obstacle to apply this inverse-free algorithm to the decoders for AG codes; we have to mention the existence of unknown syndromes, namely, the lack of syndrome values to decode errors whose Hamming weights are less than or equal to even the basic $\lfloor (d_G - 1)/2 \rfloor$, where d_G is the Goppa (designed) minimum distance. Feng and Rao’s paper [3] originally proposed majority logic scheme to determine unknown syndromes in the decoding up to $\lfloor (d_{FR} - 1)/2 \rfloor$, where d_{FR} is their designed minimum distance $\geq d_G$. In the sequel, Sakata *et al.* [26] and independently Kötter [7] modified and applied Feng–Rao’s method to their decoding algorithm. If the divisions of the finite field are removed from BMS algorithm, one cannot execute the determination of unknown syndromes because of breaking the generation of candidate values of unknown syndromes for majority voting. Unfortunately, the elimination of finite-field divisions seemed to be a difficult problem in

Manuscript received April 10, 2007. This work was partly supported by the Academic Frontier Project for Future Data Storage Materials Research by the Japanese Ministry of Education, Culture, Sports, Science and Technology (1999–2008). The material in this paper was presented in part at Hawaii, IEICE and SITA Joint Conference on Information Theory, Hawaii, May 2005, and at IEEE International Symposium on Information Theory, Seattle, July 2006.

H. Matsui and S. Mita are with the Department of Electronics and Information Science, Toyota Technological Institute, Hisakata 2–12–1, Tenpaku-ku, Nagoya 468–8511, Japan (e-mail: hmatsui@toyota-ti.ac.jp; smita@toyota-ti.ac.jp).

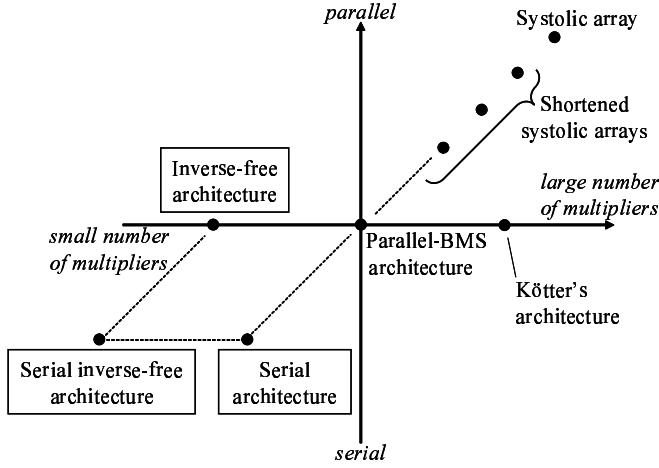


Fig. 1. Map of various error-locator architectures implementing BMS (or equivalent) algorithm for decoding codes on algebraic curves.

this regard. For this reason, no inverse-free algorithm for AG codes has been proposed until now.

In this research, we effectively overcome this difficulty. Namely, we decode such codes with the only known syndrome values from received code-words. So far the type and amount of errors that could be corrected if one does not determine unknown syndromes have not been clear; the well-known fact up to $\lfloor (d_G - g - 1)/2 \rfloor$ in Peterson-type algorithm [6], where g is the genus of underlying algebraic curve, is not available for our case of BMS algorithm. We confirm that a class of generic errors [12][23] (independent errors in [5]) can be corrected up to $\lfloor (d_{FR} - a)/2 \rfloor$ only with syndromes from received words, where a is the minimal pole order of underlying algebraic curve: $a = 2$ for elliptic curves over arbitrary finite fields and $a = 16$ for Hermitian curve over $GF(2^8)$. Furthermore, we successfully obtain the approximate ratio $(q - 1)/q$ of the generic errors to all errors in the application of Gröbner-basis theory, where q is the number of elements in the finite field. It means that we can decode most of the errors without majority logic scheme and voting. Thus we can realize not only inverse-free error-locator architectures for AG codes but also avoiding complicated procedure and transmission of voting data among parts of decoders. Our method is applicable to all former architectures, and is not a go-back to the past but a real solution to construct decoders with feasible circuit-scale.

Recently, the BMS algorithm has become more important not only in decoding codes on algebraic curves but also in algebraic soft-decision decoding [8] of RS codes. Sakata *et al.* [22][28] applied the BMS algorithm to the polynomial interpolation in Sudan and Guruswami–Sudan algorithms [4][29] for RS codes and codes on algebraic curves. Lee and O’Sullivan [9][10] applied the Gröbner-basis theory of modules, which is related to the BMS algorithm, to soft-decision decoding of RS codes. Our method can be expected to help further structural analysis of these methods.

The rest of this paper is organized as follows. In Section II, we prepare notations, and define codes on algebraic curves.

In Section III, we propose an inverse-free BMS algorithm, and state the main theorem for output of the algorithm. In the next three sections, we describe three types of small-scale error-locator architectures, i.e., *inverse-free*, *serial*, and *serial inverse-free architectures*; the mutual relations among them and past architectures are depicted in Fig. 1. In Section IV, we describe the inverse-free architecture, and divide it into three subsections: Subsection IV-A is an overview, Subsection IV-B deals with the technique for avoiding the determination of unknown syndromes, and Subsection IV-C is numerical simulation. In Section V, we describe the serial architecture using parallel BMS algorithm. In Section VI, we describe the serial inverse-free architectures combined with the previous methods. In Section VII, we estimate the total running time and the numbers of finite-field calculators for three and past architectures. Finally, in Section VIII, we state our conclusions. In the appendices, we prove the basics of BMS algorithm, the property of generic errors, and the main theorem of proposed algorithm.

II. PRELIMINARIES

In this paper, we consider one-point algebraic-geometric codes on non-singular plane curves over a finite field $K := \mathbb{F}_q$, in particular \otimes -type codes (not L -type). Let \mathbb{Z}_0 be the set of non-negative integers, and let $a, b \in \mathbb{Z}_0$ be $0 < a \leq b$ and $\gcd(a, b) = 1$. We define a C_a^b curve \mathcal{X} by an equation

$$D(x, y) := y^a + ex^b + \sum_{\substack{(n_1, n_2) \in \mathbb{Z}_0^2 \\ n_1 a + n_2 b < ab}} \chi_{(n_1, n_2)} x^{n_1} y^{n_2} = 0 \quad (3)$$

over K with $e \neq 0$. Then the polynomial quotient ring $K[\mathcal{X}] := K[x, y]/(D(x, y))$ consists of all the algebraic functions having no poles except at the unique infinite point P_∞ . Let $\{P_j\}_{1 \leq j \leq n}$ be a set of n K -rational points except P_∞ . We denote the pole order of $F \in K[\mathcal{X}]$ at P_∞ as $o(F)$. For $m \in \mathbb{Z}_0$, the K -linear subspace

$$L(mP_\infty) := \{F \in K[\mathcal{X}] \mid o(F) \leq m\} \cup \{0\}$$

has dimension $m - g + 1$, provided $m > 2g - 2$ by Riemann–Roch theorem, which we assume for simplicity in this paper. Our code $\mathcal{C}(m)$ is defined as

$$\mathcal{C}(m) := \left\{ (c_j) \in K^n \mid \sum_{j=1}^n c_j F(P_j) = 0, \forall F \in L(mP_\infty) \right\}.$$

As shown in [20][21], the class of C_a^b curves is sufficiently wide and contains almost all well-known plane algebraic curves that have many K -rational points such as Hermitian codes. Although Miura in [21] defined a more general class $rC_a^{b,d}$ including the Klein’s quartic curve, we consider mainly C_a^b for simplicity.

Throughout this paper, we denote t as the number of correctable errors. Given a received word $(r_j) = (c_j) + (e_j)$, where $e_j \neq 0 \Leftrightarrow j \in \{j_1, \dots, j_t\}$ corresponding to a set of error-locations $\mathcal{E} = \{P_{j_\gamma}\}_{1 \leq \gamma \leq t}$, we need to find a Gröbner basis [2] of the error-locator ideal

$$I(\mathcal{E}) := \{F \in K[\mathcal{X}] \mid F(P_{j_\gamma}) = 0 \text{ for } \forall P_{j_\gamma} \in \mathcal{E}\}.$$

0	2	4	6	8
3	5	7	9	11
6	8	10	12	14
9	11	13	15	
12	14			
15				

0	2	4
3	5	7
6	8	10
9	11	13
12	14	
15		

2	4	6
5	7	9
8	10	12
11	13	15
14		

4	6	8
7	9	11
10	12	14
13	15	

Fig. 2. Pole orders on $\Phi(5, 15)$ defined by $o(n) := 3n_1 + 2n_2$, and pole orders on $\Phi^{(0)}(3, 15)$, $\Phi^{(1)}(3, 15)$, $\Phi^{(2)}(3, 15)$. The values in shaded boxes correspond to monomials of the form $x^{n_1}y^{n_2}$ not contained in $L(15P_{(0;0;1)})$ of Klein's quartic curve $x^3y + y^3 + x = 0$ over $\text{GF}(2^3)$ (cf. later section V).

Then we can obtain \mathcal{E} as the set $\subset \{P_j\}_{1 \leq j \leq n}$ of common zeros of all the polynomials in the Gröbner basis.

For $A \in \mathbb{Z}_0$ and $0 \leq i < a$, let

$$\Phi^{(i)}(A) := \{n = (n_1, n_2) \in \mathbb{Z}_0^2 \mid i \leq n_2 < i + A\}$$

and $\Phi(A) := \Phi^{(0)}(A)$. Moreover, for $A' \in \mathbb{Z}_0$, let

$$\Phi^{(i)}(A, A') := \{n \in \Phi^{(i)}(A) \mid o(n) \leq A'\}$$

and $\Phi(A, A') := \Phi^{(0)}(A, A')$. Fig. 2 illustrates $\Phi(2a - 1, A')$ and $\Phi^{(i)}(a, A')$ for $A' = 15$ and $(a, b) = (3, 2)$; although we defined as $a \leq b$, it must be generalized into $a > b$ in the case of well-known Klein's quartic curve, which is one of the important examples not contained in \mathcal{C}_a^b curves; we will also take up codes on this curve later in section V. We note that $o(n) \neq o(n')$ if and only if $n \neq n'$ for $n, n' \in \Phi^{(i)}(a)$, and this is false for $\Phi(2a - 1)$. Thus $F \in K[\mathcal{X}]$ is uniquely expressed as

$$F(x, y) = \sum_{n \in \Phi(a, o(F))} F_n x^{n_1} y^{n_2}. \quad (4)$$

We denote $x^{n_1}y^{n_2}$ by z^n and define $o(n) := o(z^n) = n_1a + n_2b$, where $o(\cdot)$ is defined on both \mathbb{Z}_0^2 and $K[\mathcal{X}]$; we remember that $o(F) = \max\{o(n) \mid F_n \neq 0\}$.

From a given received word (r_j) , we calculate syndrome values $\{u_l\}$ for $l \in \Phi(2a - 1, m)$ by $u_l = \sum_{j=1}^n r_j z^l(P_j)$, where we have $u_l = \sum_{\gamma=1}^t e_{j_\gamma} z^l(P_{j_\gamma})$ by the definition of $\mathcal{C}(m)$. Our aim is to find $I(\mathcal{E})$ and (e_j) with $\{u_l\}$.

III. INVERSE-FREE BMS ALGORITHM

We continue to prepare notations to describe the algorithm. The standard partial order \leq on \mathbb{Z}_0^2 is defined as follows: for $n = (n_1, n_2)$ and $n' = (n'_1, n'_2) \in \mathbb{Z}_0^2$, $n \leq n' \Leftrightarrow n_1 \leq n'_1$ and $n_2 \leq n'_2$. For $l \in \Phi(a, A')$, let $l^{(i)} \in \Phi^{(i)}(a, A')$ be $o(l^{(i)}) = o(l)$ if there exists such an $l^{(i)}$ for l and i . Then $l^{(i)}$ is uniquely determined for each l and i if it exists. Note that $l^{(0)} = l$ from its definition. Table I illustrates $l^{(i)} \in \Phi^{(i)}(3, 15)$ for $(a, b) = (3, 2)$, where “*” indicates the nonexistence of $l^{(i)}$ from a gap-number in $o(\Phi^{(i)}(a))$.

Before the description of the algorithm, we introduce the important index \bar{i} for $0 \leq i < a$ for updating in the algorithm. For $0 \leq i < a$ and $N \in \mathbb{Z}_0$, we define a unique integer $0 \leq \bar{i} < a$ by $\bar{i} \equiv b^{-1}N - i \pmod{a}$, where the integer $0 \leq b^{-1} < a$ is defined by $bb^{-1} \equiv 1 \pmod{a}$. If there is $l^{(i)} = (l_1^{(i)}, l_2^{(i)}) \in \Phi^{(i)}(a)$ with $N = o(l^{(i)})$, then $\bar{i} = l_2^{(i)} - i$ since $l_2^{(i)} \equiv b^{-1}N \pmod{a}$. Note that $\bar{i} = i$, and that $l^{(i)}$ exists if and only if $l^{(\bar{i})}$ exists with $l^{(i)} = l^{(\bar{i})}$.

TABLE I
VALUES OF $l^{(i)} = (l_1^{(i)}, l_2^{(i)}) \in \Phi^{(i)}(3, 15)$ WITH $o(l^{(i)}) = N$

$l \backslash N$		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$l^{(0)}$	$l_1^{(0)}$	0	*	*	1	*	1	2	1	2	3	2	3	4	3	4	5
	$l_2^{(0)}$	0	*	*	0	*	1	0	2	1	0	2	1	0	2	1	0
$l^{(1)}$	$l_1^{(1)}$	*	*	*	*	*	1	*	1	2	*	2	3	2	3	4	3
	$l_2^{(1)}$	*	*	*	*	*	1	*	2	1	*	2	1	3	2	1	3
$l^{(2)}$	$l_1^{(2)}$	*	*	*	*	*	*	*	1	*	*	2	*	2	3	2	3
	$l_2^{(2)}$	*	*	*	*	*	*	*	2	*	*	2	*	3	2	4	3

We define *degree* $\deg(F) \in \Phi(a)$ of $F \in K[\mathcal{X}]$ uniquely by $o(\deg(F)) = o(F)$, and let $s := \deg(F)$. From now on, $\Phi(a, o(s))$ is abbreviated to $\Phi(a, s)$. Defining, for $l \in \Phi(a)$,

$$dF_l := \begin{cases} \sum_{n \in \Phi(a, s)} F_n u_{n+l^{(s_2)}-s} & \text{if } l^{(s_2)} \geq s, \\ 0 & \text{otherwise,} \end{cases} \quad (5)$$

where “otherwise” includes the vacant case of $l^{(s_2)}$, we call dF_l *discrepancy* of $F \in K[\mathcal{X}]$ at l . Let $V(u, N)$ be the set of $F \in K[\mathcal{X}]$ whose discrepancies are zero at all $l \in \Phi(a, N)$, and let $V(u, -1) := K[\mathcal{X}]$. Then, for all $N \in \mathbb{Z}_0 \cup \{-1\}$, $V(u, N)$ is an ideal in the ring $K[\mathcal{X}]$ (as proved at Proposition 1 in Appendix A). The BMS algorithm computes a Gröbner basis of $V(u, N)$ for each N , namely, a minimal polynomial ideal-basis with respect to the pole order $o(\cdot)$. We may express the basis of $V(u, N)$ for each N as a polynomials $\{F_{N+1}^{(i)}(z)\}_{0 \leq i < a}$ by (4). For sufficiently large B , we have $V(u, B) = I(\mathcal{E})$ (proved at Proposition 3 in Appendix B). Then $\{F_{B+1}^{(i)}(z)\}$ are called *error-locator polynomials*, and the set of their common zeros agrees with \mathcal{E} . Since the Goppa designed distance d_G of $\mathcal{C}(m)$ equals $m - 2g + 2$, we may set

$$m := 2t + 2g - 1 \quad \text{for the correction up to } t \text{ errors,} \quad (6)$$

and can obtain $V(u, m)$ by using $\{u_l\}_{l \in \Phi(a, m)}$.

In the following inverse-free BMS algorithm, we denote the preserved condition (P) for updating formulae as follows: (P) $\Leftrightarrow d_N^{(i)} = 0$ or $s_N^{(i)} \geq l^{(i)} - c_N^{(i)}$.

Inverse-free BMS Algorithm

Input syndrome values $\{u_l\}$ for $l \in \Phi(2a - 1, m)$.

Output error-locator polynomials $\{F_{m+1}^{(i)}(z)\}$.

In each step, the indicated procedures are carried out for all $0 \leq i < a$.

Step 0 (initializing) $N := 0$, $s_N^{(i)} := (0, i)$,
 $c_N^{(i)} := (-1, i)$, $v_N^{(i)}(Z) := \sum_{n \in \Phi(a, m)} u_n Z^{o(n)}$,
 $w_N^{(i)}(Z) := 1$, $f_N^{(i)}(Z) := 1$, $g_N^{(i)}(Z) := 0$.

Step 1 (checking discrepancy) If $l^{(i)}$ exists and $s_N^{(i)} \leq l^{(i)}$,
then $d_N^{(i)} := v_{N, N}^{(i)}$, else $d_N^{(i)} := 0$;
moreover, $e_N^{(i)} := w_{N, N}^{(i)}$.

Step 2 (N -updating)

$$s_{N+1}^{(i)} := \begin{cases} s_N^{(i)} & \text{if (P),} \\ l^{(i)} - c_N^{(\bar{i})} & \text{otherwise,} \end{cases} \quad (7)$$

$$c_{N+1}^{(\bar{i})} := \begin{cases} c_N^{(\bar{i})} & \text{if (P),} \\ l^{(i)} - s_N^{(i)} & \text{otherwise,} \end{cases} \quad (8)$$

$$f_{N+1}^{(i)} := e_N^{(\bar{i})} f_N^{(i)} - d_N^{(i)} g_N^{(\bar{i})}, \quad (9)$$

$$g_{N+1}^{(\bar{i})} := \begin{cases} Z g_N^{(\bar{i})} & \text{if (P),} \\ Z f_N^{(i)} & \text{otherwise,} \end{cases} \quad (10)$$

$$v_{N+1}^{(i)} := e_N^{(\bar{i})} v_N^{(i)} - d_N^{(i)} w_N^{(\bar{i})} \pmod{Z^N}, \quad (11)$$

$$w_{N+1}^{(\bar{i})} := \begin{cases} Z w_N^{(\bar{i})} & \text{if (P),} \\ Z v_N^{(i)} & \text{otherwise.} \end{cases} \quad (12)$$

Step 3 (checking termination) If $N < m$, then $N := N + 1$
and go to Step 1, else stop the algorithm. \square

In the formula (11), “ $\pmod{Z^N}$ ” means that $v_{N+1}^{(i)}$ is defined
by omitting the term of Z^N in $v_N^{(i)}$. Then $v_N^{(i)}$, $w_N^{(i)}$ can be
represented by

$$v_N^{(i)}(Z) = \sum_{h=N}^{m+N} v_{N,h}^{(i)} Z^h, \quad w_N^{(i)}(Z) = \sum_{h=N}^{m+N} w_{N,h}^{(i)} Z^h,$$

and $v_{N,N}^{(i)}$, $w_{N,N}^{(i)}$ are defined by these. We obtain $\{F_N^{(i)}(z)\}$
through

$$F_N^{(i)}(z) := \sum_{n \in \Phi(a, s)} f_{N, o(s-n)}^{(i)} z^n \quad \text{with} \quad s := s_N^{(i)}.$$

Then $d_N^{(i)}$ in the algorithm agrees with the discrepancy of $F_N^{(i)}$
at $o(l) = N$, i.e., $d_N^{(i)} = d(F_N^{(i)})_l$.

This inverse-free BMS algorithm is a novel version that
eliminates the inverse calculation $(d_N^{(i)})^{-1}$ from the parallel
BMS algorithm [16][27]. Compared with updating formulae
in the original algorithm, which are later quoted at (16)–(19),
we see that (9)–(12) have eliminated the use of divisions,
and in consequence have used $e_N^{(\bar{i})}$. It is possible that one
could remove the inverse calculation from the original (not
parallel) BMS algorithm if the values of $e_N^{(\bar{i})}$, which are
actually previous values of $d_N^{(i)}$, are registered to memory-
elements; in our parallel inverse-free BMS algorithm, we can
conveniently take $e_N^{(\bar{i})}$ from the coefficients of $w_N^{(\bar{i})}$ (as done
in Step 1).

The following theorem confirms that $\{F_N^{(i)}\}_{0 \leq i < a}$ is a
Gröbner basis of $V(u, N-1)$.

Theorem 1: We have $F_N^{(i)} \in V(u, N-1)$, $\deg(F_N^{(i)}) = s_N^{(i)}$,
 $s_{N,1}^{(0)} \geq s_{N,1}^{(1)} \geq \dots \geq s_{N,1}^{(a-1)}$, and

$$s_{N,1}^{(i)} = \min \left\{ \zeta_{N,1}^{(i)} \in \mathbb{Z}_0 \mid F \in V(u, N-1), \right. \\ \left. \deg(F) = (\zeta_{N,1}^{(i)}, i) \right\}. \quad \square \quad (14)$$

The proof of Theorem 1 is referred to Appendix D, in which
 $s_{N,1}^{(i)} = c_{N,1}^{(i)} + 1$ is also obtained for all N and i .

As explained at Proposition 3 in Appendix B, the integer B
is required as $B \geq 2t + 4g - 2 + a$ to correct up to t errors.
Moreover, it is well-known [3][26] that the determination of
unknown-syndrome values has to be done to proceed the loops
for $N = m+1, m+2, \dots, B$ of BMS algorithm. In our
Theorem 1, as a result of division-less, “ $F_{N,s}^{(\bar{i})} = 1$ ” is not
generally true differently from Theorem 1 of [16], and this fact
disables us from generating the candidate values of unknown
syndromes for majority voting. Therefore, in our inverse-
free BMS algorithm, we avoid the determination of unknown
syndrome, and the loops of the algorithm are proceeded *only*
for $0 \leq N \leq m$ by using the known syndrome values obtained
directly from the received word. Furthermore, we mainly
consider the error-correction of generic errors [5][23] (defined
in the next section). These techniques cause a slight decrease
in the error-correcting capability; however, as described later
in section IV-B, it does not matter in practice.

IV. INVERSE-FREE ARCHITECTURE

As the first of three kinds of architectures proposed in this
paper, we describe *inverse-free architecture*, which has the
plainest structure of the three.

A. Model structure

In this subsection, we give a direct application of the
inverse-free BMS algorithm, which corresponds to Kötter’s
architecture [7] of which inverse-calculators have been re-
placed by multipliers. To make the case clear, we describe the
architecture for elliptic codes, that is, codes on elliptic curves,
although we take the generality into account; we can employ
it for other codes on algebraic curves without difficulty.

As shown in the model Fig. 3, the coefficients of $v_N^{(i)}$, $f_N^{(i)}$
are arranged in a sequence of shift-registers, and those of $w_N^{(i)}$,
 $g_N^{(i)}$ are arranged in another sequence. It is similar to Kötter’s
architecture [7] that the proposed architecture has a -multiple
structure (i.e. a blocks) of the architecture for the Berlekamp–
Massey algorithm [1][11] of RS codes. The difference is that
 a division-calculators in the Kötter’s architecture are replaced
with a multipliers in our architecture. Moreover, while the
values of discrepancy are computed in the Kötter’s architecture
with one multiplier and a shift-register according to definition
(5), our architecture derives the values from the coefficients of
 $v_N^{(i)}$ with *discrepancy registers* and reduces the one multiplier
for computing discrepancy.

In Fig. 3, we omit input and output terminals, and the
initial ($N = 0$) arrangement of the coefficients in polynomials
is indicated. The number of registers in one shift-register
sequence for $v_N^{(i)}$ and $f_N^{(i)}$ should be equal to the total number
of coefficients in $v_N^{(i)}$ and $f_N^{(i)}$, i.e., $m+2$ for $\mathcal{C}(m)$; although

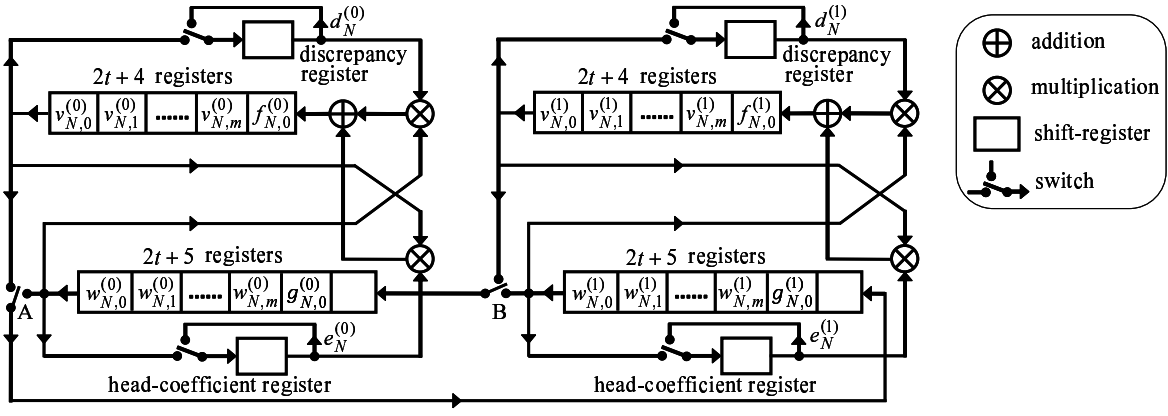


Fig. 3. Inverse-free architecture for elliptic codes, which is composed of $a = 2$ blocks exchanging $w_N^{(i)}$ and $g_N^{(i)}$.

```

1.  % initializing
2.  ll := [ 0 * 1 0 2 1 3 2 4
           * * 0 * 1 0 2 1 ];
3.  v_f_0 := [10 -1 14 12 7 5 1 8 9 0]; % 10 registers
4.  v_f_1 := [-1 -1 -1 12 -1 5 2 8 10 0]; % 10 registers
5.  w_g_0 := [0 -1 ... -1]; % 11 registers
6.  w_g_1 := [0 -1 ... -1]; % 11 registers
7.  d_0 := -1; d_1 := -1; t_0 := -1; t_1 := -1;
8.  N := -1; S := [0 0]; C := [-1 -1]; PS := S;
9.  T := [-1 -1]; M := [-1 -1];
10. % start of main clock loop
11. for clo = 0 to 11*9-1;
12.   if mod(clo,11) = 0; N := N+1; end;
13.   print [v_f_0, v_f_1, clo, w_g_0, w_g_1];
14.   if mod(clo,11) = 0;
15.     b0:=mod(clo/11,2)+1; b1:=mod(1+clo/11,2)+1;
16.     if S(1) <= ll(1,N+1); d_0 := v_f_0(1);
17.     else; d_0 := -1;
18.     end; t_0 := w_g_0(1);
19.     if S(2) <= ll(2,N+1); d_1 := v_f_1(1);
20.     else; d_1 := -1;
21.     end; t_1 := w_g_1(1);
22.   end;
23.   v_f_0_t := v_f_0(1); v_f_1_t := v_f_1(1);
24.   w_g_0_t := w_g_0(1); w_g_1_t := w_g_1(1);
25.   v_f_0(1:9) := v_f_0(2:10); v_f_1(1:9) := v_f_1(2:10);
26.   w_g_0(1:10) := w_g_0(2:11); w_g_1(1:10) := w_g_1(2:11);
27.   if mod(clo,11) = 0; v_f_0(10) := -1; v_f_1(10) := -1;
28.   else; % updating of v and f
29.     v_f_0(10) := t_0 ⊗ v_f_0_t ⊕ d_0 ⊗ w_g_0_t;
30.     v_f_1(10) := t_1 ⊗ v_f_1_t ⊕ d_1 ⊗ w_g_1_t;
31.   end;
32.   if mod(clo,11) = 0; % updating of S and C
33.     if d_0 < 0 or S(1) >= ll(1,N+1) - C(b0);
34.       ns := S(1); nc := C(b0);
35.     else; ns := ll(1,N+1) - C(b0); nc := ll(1,N+1) - S(1);
36.       T(b0) := S(1); M(b0) := N;
37.     end; S(1) := ns; C(b0) := nc;
38.     if d_1 < 0 or S(2) >= ll(2,N+1) - C(b1);
39.       ns := S(2); nc := C(b1);
40.     else; ns := ll(2,N+1) - C(b1); nc := ll(2,N+1) - S(2);
41.       T(b1) := S(2); M(b1) := N;
42.     end; S(2) := ns; C(b1) := nc;
43.   end;
44.   % updating of w and g
45.   if 8-N <= mod(clo,11) <= 8-M(b0) and N<8;
46.     w_g_1(11) := -1;
47.   elseif S(b0) = PS(b0); w_g_1(11) := w_g_0_t;
48.   else; w_g_1(11) := v_f_0_t; end;
49.   if 8-N <= mod(clo,11) <= 8-M(b1) and N<8;
50.     w_g_0(11) := -1;
51.   elseif S(b1) = PS(b1); w_g_0(11) := w_g_1_t;
52.   else; w_g_0(11) := v_f_1_t; end;
53.   if mod(clo,11) = 10; PS := S; end;
54. end; % end of main clock loop

```

Fig. 4. Program simulating the inverse-free architecture for (24, 16, 8) elliptic code $C(8)$ over $GF(2^4)$ with three-error correction.

it might seem that there is no space for $f_N^{(i)}$, it is made by shortening and shifting of $v_N^{(i)}$ as N is increased. On the other hand, the number of shift-registers required for $w_N^{(i)}$ and $g_N^{(i)}$ is one more than that for $v_N^{(i)}$ and $f_N^{(i)}$ because of the structure of parallel BMS algorithm, and should be $m + 3$.

If $N \equiv 0 \pmod{m+3}$, the switches in the discrepancy registers are closed downward to obtain the values of discrepancy $v_{N,N}^{(i)} = d_N^{(i)}$, and if $N \not\equiv 0 \pmod{m+3}$, they are closed upward to output the values of discrepancy at each clock. The head-coefficient registers work similarly to the discrepancy registers, and output the values of the head coefficient $w_{N,N}^{(i)} = e_N^{(i)}$ of $w_N^{(i)}$. The coefficients of $w_N^{(i)}$ and $g_N^{(i)}$ are transferred from the block of $v_N^{(i)}$ to that of $v_{N+1}^{(i)}$ (i for $N+1$). The switches A and B work according to the preserving

or updating of $w_N^{(i)}$ and $g_N^{(i)}$, i.e., “(P)” or “otherwise” in (10) and (12).

Thus, one may only perform simple additions and multiplications for the values in the shift-register sequences for $v_N^{(i)}$ and $f_N^{(i)}$ to update them. On the other hand, as for $w_N^{(i)}$ and $g_N^{(i)}$, one must not only perform additions and multiplications but also set register-values to zero, or else old disused values corrupt $v_N^{(i)}$ and $f_N^{(i)}$. We describe this procedure in a later subsection IV-C.

This inverse-free architecture has an a -multiple structure closer to Kötter’s than to the latter two architectures, and has been changed to division-free and parallel in the sense of using two types of polynomials, $v_N^{(i)}$ and $w_N^{(i)}$, to compute discrepancy. We see in Section VII that the total number of

TABLE II
VALUES OF REGISTERS IN FOUR SHIFT-REGISTER SEQUENCES, DISCREPANCY $d_N^{(i)}$, AND $s_{N,1}^{(i)}$ IN THE INVERSE-FREE ARCHITECTURE.

v_f_0 for $v_N^{(0)}$ and $f_N^{(0)}$										v_f_1 for $v_N^{(1)}$ and $f_N^{(1)}$										clo	w_g_0 for $w_N^{(0)}$ and $g_N^{(0)}$											w_g_1 for $w_N^{(1)}$ and $g_N^{(1)}$																
1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10		0	1	2	3	4	5	6	7	8	9	10	11	1	2	3	4	5	6	7	8	9	10	11					
10	-1	14	12	7	5	1	8	9	0	-1	-1	-1	12	-1	5	2	8	10	0	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1				
-1	14	12	7	5	1	8	9	0	-1	-1	-1	12	-1	5	2	8	10	0	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	10				
14	12	7	5	1	8	9	0	-1	-1	-1	12	-1	5	2	8	10	0	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	10	-1				
4	9	1	5	-1	9	-1	14	-1	-1	10	10	11	4	2	-1	-1	-1	-1	-1	66	14	12	1	-1	-1	10	-1	14	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1			
9	1	5	-1	9	-1	14	-1	-1	-1	10	11	4	2	-1	-1	-1	-1	-1	-1	67	12	1	-1	-1	10	-1	14	-1	-1	-1	10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	14			
1	5	-1	9	-1	14	-1	-1	-1	-1	10	11	4	2	-1	-1	-1	-1	-1	-1	68	1	-1	-1	10	-1	14	-1	-1	-1	-1	10	10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	14	12		
5	-1	9	-1	14	-1	-1	-1	-1	-1	10	10	4	2	-1	-1	-1	-1	-1	-1	69	-1	-1	10	-1	14	-1	-1	-1	-1	-1	10	10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	14	12	-1		
-1	9	-1	14	-1	-1	-1	-1	-1	-1	10	10	4	2	-1	-1	-1	-1	-1	-1	70	-1	10	-1	14	-1	-1	-1	-1	-1	10	10	-1	4	-1	-1	-1	-1	-1	-1	-1	-1	-1	14	12	-1	-1		
9	-1	14	-1	-1	-1	-1	-1	-1	-1	10	10	4	-1	-1	-1	-1	-1	-1	-1	71	10	-1	14	-1	-1	-1	-1	-1	-1	10	10	-1	4	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1			
-1	14	-1	-1	-1	-1	-1	-1	-1	-1	10	10	4	-1	6	-1	-1	-1	-1	-1	72	-1	14	-1	-1	-1	-1	-1	-1	-1	-1	10	10	-1	4	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	10		
14	-1	-1	-1	-1	-1	-1	-1	-1	-1	10	10	4	-1	6	-1	-1	-1	-1	-1	73	14	-1	-1	-1	-1	-1	-1	-1	-1	-1	10	10	-1	4	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	10		
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	10	10	4	-1	6	-1	8	-1	-1	-1	74	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	10	10	-1	4	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	14	
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	10	10	4	-1	6	-1	8	-1	-1	-1	75	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	10	10	-1	4	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	14	
-1	10	10	4	-1	6	-1	8	-1	-1	-1	10	11	4	2	-1	-1	-1	-1	-1	76	-1	10	10	-1	4	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	14	
10	10	4	-1	6	-1	8	-1	-1	-1	-1	10	11	4	2	-1	-1	-1	-1	-1	77	10	10	-1	4	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	14	
-1	-1	13	13	12	-1	2	-1	-1	-1	-1	-1	13	11	10	2	-1	4	-1	-1	97	-1	-1	10	-1	-1	4	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	14	
-1	13	13	12	-1	2	-1	-1	-1	-1	-1	-1	13	11	10	2	-1	4	-1	-1	98	-1	10	-1	-1	4	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	14
13	13	12	-1	2	-1	-1	-1	-1	-1	-1	-1	13	11	10	2	-1	4	-1	-1	out	10	-1	-1	4	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	14

$d_N^{(i)}$

i	N	0	1	2	3	4	5	6	7	8
0	10	-1	14	-1	14	-1	4	10	-1	
1	-1	-1	-1	7	-1	7	10	10	6	

$s_{N,1}^{(i)}$

i	N	0	1	2	3	4	5	6	7	8	out
0	0	1	1	1	1	2	2	2	2	2	
1	0	0	0	0	0	0	0	0	1	1	1

shift-registers in our architecture is nearly the same as that in Kötter's, i.e., the additional polynomials do not contribute essentially to the total number of registers.

B. Decoding of generic errors

To implement the inverse-free algorithm effectively, we concentrate on decoding generic t -errors [5][23], for which the degree $s_N^{(i)}$ of error-locator polynomials is characterized by $o(s_N^{(i)}) \leq t + g - 1 + a$, while in general we have $o(s_N^{(i)}) \leq t + 2g - 1 + a$. In other word, the error-location \mathcal{E} is generic if and only if so-called *delta set* $\{l \in \Phi(a) \mid l \leq s_N^{(i_2)}\}$ of error-locator polynomials corresponds to the first t non-gaps in $o(\Phi(s))$. Then the loops of BMS algorithm are required for $0 \leq N \leq m + a - 1$ to obtain the error-locator polynomials for generic t -errors, while in general $0 \leq N \leq m + 2g - 1 + a$ for all errors; these facts are proved in Appendix C. Thus we see that $(t - \lceil(a-1)/2\rceil)$ errors are corrected in $\mathcal{C}(m)$ after N -updating for $0 \leq N \leq m$. The merits of this method are not only that it is inverse-free and there is no majority logic [3] but also that there are fewer loops of the BMS algorithm; we can cut it down to $2g - 1$ loops. Furthermore, this method can also be applied to Kötter's and systolic-array architectures [16].

There are two drawbacks to this method. The first is that non-generic errors cannot be corrected. Since generic or non-generic is also defined by whether a matrix determinant $\neq 0$ or not (as shown in Appendix C), the ratio of generic errors to all errors is estimated at $(q-1)/q$, under the hypothesis for the randomness of values $\{z^l(P_j)\}$ (which is supported by numerical tests [12]). As for a practical size $q = 2^8$, the ratio is equal to $255/256 = 0.9960 \dots$. Moreover, for errors less than t , the percentage of correctable errors increases

since $o(s_N^{(i)})$ s decrease. Thus we have less effect of this drawback. The second is that the number of correctable errors is decreased $\lceil(a-1)/2\rceil$ for t -error correctable codes $\mathcal{C}(m)$. This corresponds to $t-1$ errors for all elliptic codes, and $t-8$ errors for Hermitian codes over \mathbb{F}_{2^8} . However, this has no serious effect on practical function; we might choose $\mathcal{C}(m+a-1)$ to correct t errors, and the remaining error-correcting capability is available for error-detection up to $t + \lfloor(a-1)/2\rfloor$ errors. In the next subsection, we demonstrate the decoding of $\mathcal{C}(m)$ with $m := m+1$ (i.e. $a=2$) for t -error correction in codes on elliptic curves.

C. Simulation and numerical example

In this subsection, we focus on an elliptic code, especially on the elliptic curve defined by the equation $y^2 + y = x^3 + x$ over $K := \mathbb{F}_{16}$, and simulate a decoder for it. This curve has 25 K -rational points equal to the Hasse-Weil bound with genus one, and we obtain code $\mathcal{C}(m)$ of length 24.

We choose a primitive element α of K satisfying $\alpha^4 + \alpha = 1$, and represent each non-zero element of K as the number of powers of α . Moreover, we represent zero in K as -1 ; note that, e.g., 0 and -1 mean $1 = \alpha^0$ and 0, respectively. Let the set of error-locations $\mathcal{E} := \{(x, y) = (3, 7), (9, 11), (14, 4)\}$, and let the error-values be 6, 8, 11, respectively.

In Fig. 4, we provide a brief description of MATLAB m-file program for our architecture, where $\text{mod}(x, Y)$ returns the smallest non-negative integer satisfying $x \equiv \text{mod}(x, Y) \pmod{Y}$. Comments are written next to “%.” At line 2, $\text{ll}(1+i, 1+N)$, which corresponds to the $(1+i, 1+N)$ -th component of matrix ll in MATLAB m-file notations, defines $l_1^{(i)}$ with $N = o(l^{(i)})$ of $l^{(i)} \in \Phi^{(i)}(2, 8)$ to decode 3 errors in $\mathcal{C}(8)$ with $m = 8$. In the case $l_1^{(i)} = *$ in ll , the logical sentences at lines 16 and 19 are regarded to be false.

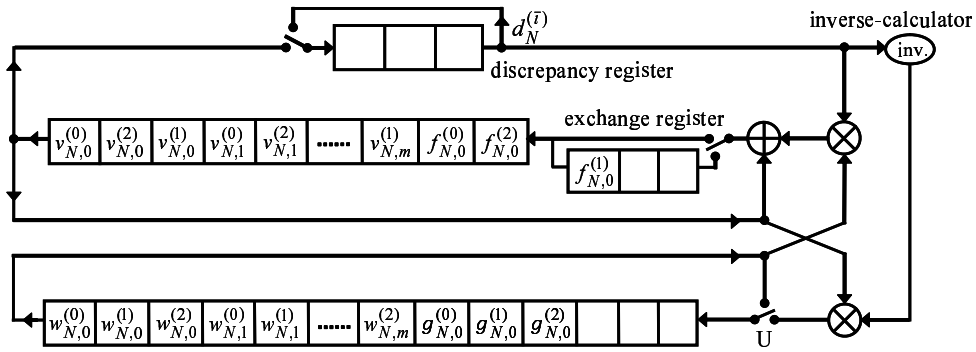


Fig. 5. Serial architecture for Klein-quartic codes, which has a single structure with serially-arranged coefficients.

In the case of elliptic codes $\mathcal{C}(m+1)$, the number of registers for $v_N^{(i)}$ and $f_N^{(i)}$ should be $(m+1) + 2 = 2t + 4$ by (6), and that for $w_N^{(i)}$ and $g_N^{(i)}$ should be $2t + 5$, as in lines 3–6 for $t = 3$. At line 15, the value b_0 (resp. b_1) corresponds to \bar{t} at N for $i = 0$ (resp. $i = 1$). At lines 25 and 26, the shift-register values are shifted to the neighbors, and, e.g., “ $v_f_0(1:9):=v_f_0(2:10)$ ” indicates the shifts of nine values $v_f_0(1):=v_f_0(2), \dots, v_f_0(9):=v_f_0(10)$, where $v_f_0(n)$ corresponds to the n -th component of v_f_0 .

Table II shows that our architecture outputs the error-locator polynomials $\{F_{m+1}^{(i)}(z)\}$ and the auxiliary polynomials $\{G_{m+1}^{(i)}(z)\}$ for \mathcal{E} . The top of Table II indicates the indexes of registers of four shift-register sequences. The center column indicates the values of “clo” in the program, which corresponds to the underlying clock of the architecture. The values of discrepancy $d_N^{(i)}$ are indicated at the left bottom of Table II, where “ -1 ” indicates the state that $l^{(i)}$ does not exist or $s_{N,1}^{(i)} > l_1^{(i)}$. The values of discrepancy $d_N^{(i)}$ are obtained at $\text{clo} = 11N$ from $v_f_0(1)$ or $v_f_1(1)$ if $s_{N,1}^{(i)} \leq l_1^{(i)}$. The values of $s_N^{(i)}$ are indicated at the right bottom of Table II.

The most difficult point in the program is that suitable register values must be settled to -1 at the lines 45 and 49 for not changing the coefficients of $f_N^{(i)}$. Let $t_N^{(i)} := \deg(G_N^{(i)}(z))$ and $M^{(i)}$ be the value of N at which the last updating of $G_N^{(i)}$ occurred; we have $t_N^{(i)} = s_{M^{(i)}}^{(i)}$ with \bar{t} at $M^{(i)}$, and have $t_{N,1}^{(i)} = T(1+i)$, $M^{(i)} = M(1+i)$ in the program. Then, we claim that $g_{N,N-M^{(i)}}^{(i)}$, that is, the head coefficient of

$$g_N^{(i)} = \sum_{h=N-M^{(i)}}^{o(t_N^{(i)})+N-M^{(i)}} g_{N,h}^{(i)} Z^h$$

is located at the $(10 - M^{(i)})$ -th register of w_g_0 or w_g_1 according to $\bar{t} = 0$ or 1 if $\text{mod}(\text{clo}, 11) = 0$. For example, if $\text{clo} = 66$ and $N = 6$, we can see from $s_{N,1}^{(i)}$ in Table II that $M^{(0)} = 4$. Then $g_{6,2}^{(0)} = \alpha^{10}$ is in $w_g_0(6)$. As another example, if $\text{clo} = 77$ and $N = 7$, we can see that $M^{(1)} = 6$, and then $g_{7,1}^{(1)} = \alpha^4$ is in $w_g_0(4)$.

Noting that the value in $w_g_0(j)$ at $\text{mod}(\text{clo}, 11) = 0$ is the shifted value at $\text{mod}(\text{clo}, 11) = j - 1$, e.g., $w_g_0(11) := w_g_1(1)$, we obtain the upper and lower conditions of $w_g_0(11)$ and $w_g_1(11) := -1$ at lines 45 and 49, since each

$N + 1 - M^{(i)}$ value of $w_g_0(j)$ and $w_g_1(j)$ for $j = 9 - N, 9 - N + 2, \dots, 9 - M^{(i)}$ must be $-1 \pmod{11} = 0$ in each $w_N^{(i)}$. The condition “ $N < 8$ ” is required to obtain the values of $e_9^{(i)} := w_{9,9}^{(i)}$ for error-evaluation (stated below).

Thus, the Gröbner basis $\{F_9^{(0)} = \alpha^{13}x^2 + \alpha^{13}y + \alpha^{12}x + \alpha^2, F_9^{(1)} = \alpha^{13}xy + \alpha^{11}x^2 + \alpha^{10}y + \alpha^2x + \alpha^4\}$ of ideal $I(\mathcal{E})$ has been obtained together with the auxiliary polynomials $\{G_9^{(0)} = \alpha^{10}x + \alpha^{14}, G_9^{(1)} = \alpha^4y + \alpha^2x\}$. We obtain the set \mathcal{E} of error-locations through the Chien search, and obtain each error-value by O’Sullivan’s formula [24]

$$e_j = \left(\sum_{0 \leq i < a} \frac{F_{m+1}^{(i)'}(P_j)}{F_{m+1,s}^{(i)}} \frac{G_{m+1}^{(i)}(P_j)}{e_{m+1}^{(i)}} \right)^{-1} \text{ for } P_j \in \mathcal{E}, \quad (15)$$

where $F_{m+1}^{(i)'}(z)$ is the formal derivative of $F_{m+1}^{(i)}(z)$ with respect to x , e.g., $y' = x^2 + 1$. Note that the divisions in this formula are independent from BMS algorithm, and are calculated by the repetitional multiplications using the multipliers in our architecture as follows.

Since we have $\beta^{-1} = \beta^{2^n-2}$ for $0 \neq \beta \in \mathbb{F}_{2^n}$, and have $a_n = 2^n - 1$ for the sequence defined by $a_1 := 1$ and $a_{n+1} := 2a_n + 1$, we see that the calculation of β^{-1} consists of $(n-2)$ multiplications of β and $(n-1)$ squares, and the total is $(2n-3)$ multiplications in \mathbb{F}_{2^n} . Thus we can say that our architecture eliminates a inverse-calculators, each of which corresponds to $(2n-3)$ multipliers, with $\lfloor \frac{a-1}{2} \rfloor$ slight drop of error-correction capability for $\mathcal{C}(m+a-1)$.

V. SERIAL ARCHITECTURE

As the second architecture, we describe *serial architecture* [13], which has a different structure from Kötter’s and the preceding ones. In this section, we focus on well-known codes on Klein’s quartic curve over $K := \mathbb{F}_8$, and simulate a decoder for it. Many articles so far have treated codes on this curve as examples.

Klein’s quartic curve is defined by equation $X^3Y + Y^3Z + Z^3X = 0$ in projective plane $\mathbb{P}^2 = \{(X : Y : Z)\}$, which causes $y^3x + x^3 + y = 0$ by $(x, y) := (Y/Z, X/Z)$ in the affine form, and has the same number of K -rational points as Hasse–Weil–Serre upper bound 24 with genus 3. We denote K -rational points $(X : Y : Z) = (1 : 0 : 0)$ and $(0 : 1 : 0)$ as $P_{(1:0:0)}$ and $P_{(0:1:0)}$, and other 22 points as the values

Fig. 6. Program simulating the serial architecture for $(23, 10, 11)$ code $\mathcal{C}(15)$ on Klein's quartic over $\text{GF}(2^3)$ with four-error correction.

We can see that the exchange register works like a shift-register, since the order-changing has been finished at $\text{clo} = 9$

TABLE III
VALUES OF REGISTERS IN TWO SHIFT-REGISTER SEQUENCES, DISCREPANCY $d_N^{(i)}$, AND $s_{N,1}^{(i)}$ IN THE SERIAL ARCHITECTURE.

v_f_r for $v_N^{(i)}$ and $f_N^{(i)}$		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	...	50		
clo	0	5	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	6	-1	6	3	-1	-1	2	2	-1	3	-1	3	4	-1	-1	6	6	6	0	-1	0	4	3	3	6		0		
...
594	3	3	-1	5	-1	4	4	3	3	3	2	0	6	6	6	0	0	0	0	-1	-1	4	2	-1	-1	-1	2	-1	-1	-1	1	2	-1	-1	-1	0	-1	-1	-1	0	-1		-1		
595	3	-1	5	-1	4	4	3	3	3	2	0	6	6	6	6	0	0	0	0	-1	-1	4	2	-1	-1	-1	2	-1	-1	-1	1	2	-1	-1	-1	0	-1	-1	-1	0	-1	-1		-1	
596	-1	5	-1	4	4	3	3	3	2	0	6	6	6	6	0	0	0	0	-1	-1	4	2	-1	-1	-1	2	-1	-1	-1	1	2	-1	-1	-1	0	-1	-1	-1	0	-1	-1	-1		-1	
...
648	-1	4	4	3	3	0	2	0	2	6	6	5	0	0	0	0	-1	4	-1	-1	-1	2	2	-1	-1	-1	1	-1	-1	-1	3	0	-1	-1	-1	0	-1	-1	-1	0	-1		-1		
649	4	4	3	3	0	2	0	2	6	6	5	0	0	0	0	-1	4	-1	-1	-1	2	2	-1	-1	-1	1	-1	-1	-1	3	0	-1	-1	-1	0	-1	-1	-1	0	-1	-1		-1		
650	4	3	3	0	2	0	2	6	6	5	0	0	0	0	-1	4	-1	-1	-1	2	2	-1	-1	-1	1	-1	-1	-1	3	0	-1	-1	-1	0	-1	-1	-1	0	-1	-1	-1		-1		
...
out	0	0	0	2	-1	-1	0	1	-1	-1	6	0	6	-1	3	-1	2	-1	-1	-1	2	5	-1	-1	-1	6	-1	-1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1		-1	

w_g_r for $w_N^{(i)}$ and $g_N^{(i)}$		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	...	54				
clo	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1		
...
594	0	-1	-1	4	-1	-1	2	-1	-1	2	-1	-1	5	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	4		-1			
595	-1	-1	4	-1	-1	2	-1	-1	2	-1	-1	5	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	4	-1	0			
596	-1	4	-1	-1	2	-1	-1	2	-1	-1	5	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	4	-1	-1	0				
...	
648	0	0	-1	2	-1	-1	1	0	-1	0	6	-1	-1	-1	-1	4	4	-1	-1	-1	-1	6	-1	-1	-1	6	-1	-1	-1	-1	6	-1	-1	-1	4	-1	-1	-1	-1	-1	-1	-1		-1			
649	0	-1	2	-1	-1	1	0	-1	0	6	-1	-1	-1	-1	4	4	-1	-1	-1	-1	6	-1	-1	-1	6	-1	-1	-1	-1	6	-1	-1	-1	4	-1	-1	-1	-1	-1	-1	-1		0				
650	-1	2	-1	-1	1	0	-1	0	6	-1	-1	-1	-1	4	4	-1	-1	-1	-1	6	-1	-1	-1	6	-1	-1	-1	-1	6	-1	-1	-1	4	-1	-1	-1	-1	-1	-1	-1		0					
...	
out	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	4	4	-1	-1	-1	-1	6	-1	-1	-1	6	-1	-1	-1	-1	6	-1	-1	-1	4	-1	-1	-1	-1	-1	-1	-1		-1			

$d_N^{(i)}$	i	N	0	...	3	...	5	6	7	8	9	10	11	12	13	14	15
	0	5	-1	0	-1	6	5	-1	0	-1	3	3	-1	3	2	6	
	1	-1	-1	-1	-1	6	-1	-1	0	-1	-1	3	4	0	2	-1	
	2	-1	-1	-1	-1	-1	-1	2	-1	-1	3	-1	4	3	-1	-1	

$s_{N,1}^{(i)}$	i	N	0	1~6	7~11	12~15	output
	0	0	1	2	3	3	
	1	1	1	1	2	2	
	2	1	1	1	1	1	

and the omission by $\text{mod } Z^N$ in (11) has been done after a more clo's.

The number of registers in one shift-register sequence for $v_N^{(i)}$ s and $f_N^{(i)}$ s should be equal to the total number of coefficients minus one, i.e., $3(m+2) - 1$ for $\mathcal{C}(m)$, and this works like $3(m+2)$ together with the exchange registers. On the other hand, $w_N^{(i)}$ s and $g_N^{(i)}$ s require a more shift-registers than $v_N^{(i)}$ s and $f_N^{(i)}$ s because of the structure of parallel BMS algorithm. Thus the number of registers for $w_N^{(i)}$ s and $g_N^{(i)}$ s should be $3(m+2) + 3$. Then $6t + 26$ and $6t + 30$ registers are required for $\mathcal{C}(m+2)$ with $m = 2t + 5$.

In Fig. 6, we describe the architecture with a MATLAB m-file program, where the notations are the same as in Fig. 4. At line 6, the values of $[s_{N,1}^{(0)}, s_{N,1}^{(1)}, s_{N,1}^{(2)}]$ and $[c_{N,1}^{(0)}, c_{N,1}^{(1)}, c_{N,1}^{(2)}]$ are initialized differently from all 0 and -1 because of the exclusion of $\{(0, 1), (0, 2)\}$ from $\Phi(3)$.

The most difficult point in the program is again that suitable register values should be settled to zero at line 40 in the successive loop for not meeting the coefficients of $f_N^{(i)}$. Since $\alpha^0 = f_{0,0}^{(0)}$ is at the 49-th register in the initial values of v_f_r , we claim that $g_{N,N-M^{(i)}}^{(i)}$ (the head coefficient of $g_N^{(i)}$) is located at the $(49 - 3M^{(i)})$ -th register of w_g_r if $\text{mod}(\text{clo}, 54) = i$. For example, if $\text{clo} = 648$ and $N = 12$, we can see from $s_{N,1}^{(i)}$ in Table III that $M^{(0)} = M^{(1)} = 11$. Then $g_{12,1}^{(0)} = g_{12,1}^{(1)} = \alpha^4$ are in $w_g_r(16)$ at $\text{clo} = 648$ and 649.

Similarly as in Subsection IV-C, we note that the value in

$w_g_r(j)$ at $\text{mod}(\text{clo}, 54) = i$ is the shifted value at $\text{mod}(\text{clo}, 54) = i + j - 1$, e.g., $w_g_r(54) := v_f_r(1)$. Moreover, since each $N + 1 - M^{(i)}$ value of $w_g_r(j)$ for $j = 46 - 3N, 46 - 3N + 3, \dots, 46 - 3M^{(i)}$ must be -1 at $\text{mod}(\text{clo}, 54) = i$ in each $w_N^{(i)}$, we obtain the upper and lower conditions of $w_g_r(54) := -1$ at line 40 as the union of

$$\begin{aligned} i = 0 &\Rightarrow j = 45 - 3N, \dots, 45 - 3M^{(0)}, \\ i = 1 &\Rightarrow j = 46 - 3N, \dots, 46 - 3M^{(1)}, \\ i = 2 &\Rightarrow j = 47 - 3N, \dots, 47 - 3M^{(2)}. \end{aligned}$$

Thus we have obtained the error-locator polynomials

$$\begin{aligned} F_{16}^{(0)} &= x^3 + x^2 + \alpha^3 xy + \alpha^2 x + \alpha, \\ F_{16}^{(1)} &= x^2 y + \alpha x^2 + \alpha^6 xy + \alpha^2 x + \alpha^6, \\ F_{16}^{(2)} &= xy^2 + \alpha^2 x^2 + xy + \alpha^6 x + \alpha^5, \end{aligned}$$

whose common zeros in the rational points decide \mathcal{E} , and the auxiliary polynomials

$$\begin{aligned} G_{16}^{(0)} &= \alpha^4 xy + \alpha^6 x + \alpha^6, \quad G_{16}^{(1)} = 0, \\ G_{16}^{(2)} &= \alpha^4 x^2 + \alpha^6 x + \alpha^4. \end{aligned}$$

Then we obtain each error-value by O'Sullivan's formula [24]

$$e_j = \left(\sum_{0 \leq i < a} F_{m+1}^{(i)}(P_j) G_{m+1}^{(i)}(P_j) \right)^{-1} \text{ for } P_j \in \mathcal{E},$$

TABLE IV
VALUES OF REGISTERS IN TWO SHIFT-REGISTER SEQUENCES, DISCREPANCY $d_N^{(i)}$, AND $s_{N,1}^{(i)}$ IN THE SERIAL INVERSE-FREE ARCHITECTURE.

clo	v_{f_r} for $v_N^{(i)}$ and $f_N^{(i)}$																																							
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	...	103		
0	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	12	-1	-1	-1		0			
⋮																																								
1904	1	6	6	1	5	6	-1	9	11	-1	-1	11	3	9	5	0	11	1	1	8	12	3	13	5	5	-1	1	2	12	11	10	12	11	8	8	11		-1		
⋮																																								
2016	6	-1	13	12	-1	-1	8	8	9	5	11	9	1	1	10	10	3	13	0	-1	-1	1	1	14	11	10	12	4	8	8	8	2	-1	-1	-1	10		-1		
⋮																																								
out	5	12	12	11	9	5	-1	-1	-1	0	-1	-1	-1	-1	4	10	9	-1	-1	8	6	2	-1	-1	-1	7	8	-1	-1	-1	4	2	14	-1	-1	1				
			37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63											
out		9	-1	-1	-1	-1	7	13	-1	-1	-1	12	-1	-1	-1	-1	2	14	-1	-1	-1	-1	-1	-1	-1	-1	-1	1												

[illegible]

$d_N^{(i)}$	i	0	4	5	8	9	10	12	13	14	15	16	17	18	19	20	21	22	23	24
	0	1	0	-1	6	-1	-1	12	14	-1	-1	-1	1	-1	-1	11	6	-1	-1	-1
	1	-1	-1	2	-1	0	12	-1	14	7	-1	-1	1	12	-1	-1	0	9	-1	-1
	2	-1	-1	-1	-1	-1	0	-1	-1	3	-1	-1	-1	6	11	-1	-1	-1	-1	-1
	3	-1	-1	-1	-1	-1	-1	-1	-1	-1	10	-1	-1	-1	-1	6	-1	-1	2	10

$s_{N,1}^{(i)}$	$i \ N$	0	1~8	9~10	11~17	18~out
	0	0	1	2	2	3
	1	0	0	0	1	2
	2	0	0	0	0	0
	3	0	0	0	0	0

We demonstrate 5-error correction, and set the error-locations $\mathcal{E} := \{(x, y) = (-1, 0), (5, 3), (9, 8), (10, 13), (12, 2)\}$, and let error values be 11, 13, 2, 12, 9, respectively.

As shown in the model Fig. 7, the serial inverse-free architecture also has the same single structure as that of RS codes. Initially, the coefficients of $v_N^{(i)}$ s and $f_N^{(i)}$ s are arranged serially in the order $i = 0, 1, 2, 3$ in a sequence of shift-registers, and those of $w_N^{(\bar{i})}$ s and $g_N^{(\bar{i})}$ s are arranged in the order $\bar{i} = 0, 3, 2, 1$ in another. This arrangement of coefficients is decided by the pair (i, \bar{i}) with $i + \bar{i} \equiv 0 \pmod{4}$, and in general for other codes on C_a^b curves, one can also arrange them in a similar manner with $i + \bar{i} \equiv 0 \pmod{a}$. Then the exchange register changes the order $i = 0, 1, 2, 3$ of $\{v_N^{(i)}, f_N^{(i)}\}$ s at $N \equiv 0 \pmod{4}$ into $i = 1, 2, 3, 0$ at $N \equiv 1, \dots, i = 3, 0, 1, 2$ at $N \equiv 3$. In general, for other codes on C_a^b curves, it changes the order of i so as to keep $i + \bar{i} \equiv b^{-1}N \pmod{a}$ as the definition of \bar{i} .

In the case of the serial inverse-free architecture, we require two other sequences of a shift-registers, *supplementary registers*, as in Fig. 7. These do not appear in the algorithm but are due to technical reasons in the architecture. For example, we can see in Table IV that the values $s_{17,1}^{(0)} = 2$ and $s_{17,1}^{(1)} = 1$ are increased to 3 and 2 at the same $N = 18$. For such cases, the supplementary registers hold the values of the head coefficients $v_{N,N}^{(i)}$ and $w_{N,N}^{(j)}$; otherwise the value $w_{N,N}$ cannot be updated to $v_{N,N}^{(i)}$.

For the same reason as the previous ones, suitable register

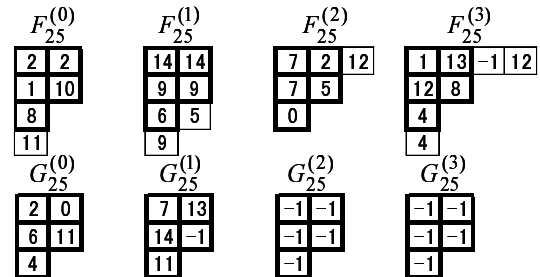


Fig. 9. Output of the serial inverse-free architecture, where polynomials are depicted on $\Phi(4, 9)$.

values should be set to zero at line 41, where the condition is derived by taking the supplementary registers into account as follows: Since $\alpha^0 = f_{N,0}^{(0)}$ is at the 101-th register in the initial values of `v_f_r` as seen in line 3, we claim that the head coefficient $g_{N,N-M^{(i)}}^{(i)}$ is located at the $(101 - 4M^{(i)})$ -th register of `w_g_r` if $\text{mod}(\text{clo}, 112) = i$. For example, if $N = 18$, we can see from $s_{N,1}^{(i)}$ in Table IV that $M^{(0)} = M^{(1)} = 17$. Then, in `w_g_r(33)`, $g_{18,1}^{(0)} = \alpha^{11}$ is at `clo` = 2016, and $g_{18,1}^{(1)} = \alpha^{11}$ is at `clo` = 2019.

Similarly as in section V, we note that the value in $w_g r(j)$ at $\text{mod}(\text{clo}, 112) = i$ is the shifted value at $\text{mod}(\text{clo}, 112) = i + j - 1 + 4$, where “+4” is caused by the supplementary four shift-registers. Moreover, since each $N + 1 - M^{(i)}$ value of $w_g r(j)$ for $j = 97 - 4N, 97 - 4N + 4, \dots, 97 - 4M^{(i)}$

must be -1 at $\text{mod}(\text{clo}, 112) = i$ in each $w_N^{(i)}$, we obtain the upper and lower conditions of $w_g_r(108) := -1$ at line 41 as the union of

$$\begin{aligned} i = 0 &\Rightarrow j = 100 - 4N, \dots, 100 - 4M^{(0)}, \\ &\vdots \\ i = 3 &\Rightarrow j = 103 - 4N, \dots, 103 - 4M^{(3)}. \end{aligned}$$

Thus, the Gröbner basis of ideal $I(\mathcal{E})$ and the auxiliary polynomials have been obtained as in Fig. 9, e.g.,

$$F_{25}^{(0)} = \alpha^{11}x^3 + \alpha^{10}xy + \alpha^8x^2 + \alpha^2y + \alpha x + \alpha^2,$$

and obtain each error-value by O'Sullivan's formula (15).

In this manner, we have constructed the smallest-scale architecture, which uses the supplementary registers differently from the others. In our example, the total number of shift-registers for polynomials is 215, while for the supplementary registers, it is 8, i.e., 3.7%. Furthermore, this percentage is decreased for larger t , and approximately $1/m$, as seen in the next section; we have, e.g., $m = 2t + 239$ for the other Hermitian codes over \mathbb{F}_{256} . Hence we can say that $2a$ shift-registers for the supplementary registers are reasonably small in the whole architecture.

VII. PERFORMANCE ESTIMATION

In this section, we estimate the numbers of multipliers, calculators for inverse, and registers, and the total running time. Although the estimation at Section IX in [16] was done with respect to the upper bound $\lambda = t + 2g - 1 + a$ of $o(s_N^{(i)})$ s, it is now convenient to estimate with respect to $m = 2t + 2g - 1$ of the code $\mathcal{C}(m)$ since we consider architectures without the determination of unknown-syndrome values.

We quote the result of the systolic array in [16]; the numbers of multipliers and calculators for inverse are $2am$ and $am/2$, respectively, as seen at the upper part of Fig.4 in [p.3866,16]. The number of registers and the total running time are $(4m + 9)a/2$ and $m + 1$, respectively.

The Kötter's architecture [7] has $3a$ multipliers, a calculators for inverse, and $a(4\lambda + 5)$ registers, where $\lambda = (m+1)/2 - 1 + a$ since we restrict correctable errors to the generic errors. The total running time takes $2(\lambda+1)(m+1) = (m+3)(m+1)$.

The serial architecture and the serial inverse-free architecture have two multipliers, and the inverse-free architecture has a times two multipliers. There is one calculator for inverse only in the serial architecture. The number of registers for these three architectures is equal to $2a$ times $m + 2$, which consists of the number of syndromes including the gaps plus one for the initial value of $f_N^{(i)}$; we ignore the contribution of the discrepancy, exchange, and supplementary registers since these are at most a few multiples of a and disappear in the order of m . The total running time for the inverse-free architecture agrees with $m + 1$ times the number of registers in the sequence for $w_N^{(i)}$ and $g_N^{(i)}$, which is equal to $(m + 1)(m + 2)$. Those for the other two agree with $a(m + 1)(m + 2)$.

We summarize these results in Table V, where we denote only the terms of the highest orders for m in the estimations. In

TABLE V
PERFORMANCE OF VARIOUS ARCHITECTURES.

Architecture	Number of \otimes	Number of inv.	Number of registers	Running time
Systolic array	$2am$	$am/2$	$2am$	m
Kötter's	$3a$	a		m^2
Parallel-BMS	$2a$	a		m^2
Inverse-free	$2a$	0		m^2
Serial	2	1		am^2
Serial inverse-free	2	0		am^2

addition, there is an architecture between Kötter's and Inverse-free that employs the parallel BMS algorithm (not inverse-free); we call this temporarily *parallel-BMS architecture* and add it to the table. For example, in the case of Hermitian codes over 2^8 -element finite field, a and m is equal to 16 and $2t + 239$, respectively. Since the numbers of registers in all architectures have an unchanged order $2am$ in Table V, we can see that these architectures have optimized their space complexity.

Then we can see in Table V that a multipliers have been reduced from Kötter's to Parallel-BMS, and that a inverse-calculators have been reduced from Parallel-BMS to Inverse-free. Both contribute to the reduction of computational complexity. It is noticed that the latter reduction has been accompanied in $\mathcal{C}(m + a - 1)$ by the slight decrease $\lfloor \frac{a-1}{2} \rfloor$ of correctable errors that is assignable to error-detection. On the other hand, two types of serial architectures have the constant numbers of finite-field calculators, and their running time takes a times longer than that of non-serial types. Thus our serializing method has provided a preferred trade-off between calculators and delay.

VIII. CONCLUSIONS

In this paper, we have proposed the inverse-free parallel BMS algorithm for error-location in decoding algebraic-geometric codes. Thus we have improved decoding bound $t \leq \lfloor (d_G - g - 1)/2 \rfloor$ in [6] based on linear system without the determination of unknown syndromes for AG codes, to $t \leq \lfloor (d_{FG} - a)/2 \rfloor$ for generic errors, where, e.g., $g = 120$ and $a = 16$ for Hermitian codes over \mathbb{F}_{2^8} . Moreover, we have constructed three kinds of error-locator architectures using our algorithm. These architectures were not implemented until the determination procedure of unknown syndromes was removed from the error-location algorithm. Our novel algorithm and architectures have a wide range of applications to Gröbner-basis schemes in various algebraic-coding situations, such as Sudan algorithm [29], Guruswami-Sudan algorithm [4], Koetter-Vardy algorithm [8], and encoding of algebraic codes [19].

We have aimed to construct our architectures with only shift-registers, switches, and finite-field calculators. The com-

position of shift-registers is superior to that of RAMs (random-access memories) in decoding speed, and moreover, our approach is useful for revealing their regularity.

We can conclude that the error-locator architectures correcting generic errors have been completed by the whole from systolic array (max. parallelism) to serial inverse-free ones (min. parallelism). These architectures enable us to fit the decoder of the codes to various sizes and speeds in many applications. It may also be concluded that our methodology, which is the direct decoding from only the received syndromes, correctly generalizes the RS-code case.

APPENDIX A

PROOF THAT $V(u, A)$ IS AN IDEAL

We first note that, by (5) and the following lemma,

$$\begin{aligned} f \in V(u, A) &\Leftrightarrow df_l = 0 \text{ for } l \in \Phi(a, A) \\ &\Leftrightarrow \sum_{n \in \Phi(a, s)} f_n u_{n+h} = 0 \text{ for } h \in \Phi(a, A - o(s)). \end{aligned} \quad (20)$$

Lemma 1: We have $\{l^{(s_2)} - s \mid l \in \Phi(a, A), l^{(s_2)} \geq s\} = \Phi(a, A - o(s))$. \square

Proof. Obviously $\{l^{(s_2)} - s \mid l \in \Phi(a, A), l^{(s_2)} \geq s\}$ equals

$$\{l - s \mid l \in \Phi^{(s_2)}(a, A), l \geq s\} = \Phi(a, A - o(s)),$$

where the last equality follows from correspondence $l - s =: h \in \Phi(a, A - o(s))$. \square

For simplicity, we denote P_j and e_j as $P_{\gamma_j} \in \mathcal{E}$ and the error-value e_{γ_j} without loss of generality. Then we convert the sum $\sum f_n u_{n+h}$ in (20) as

$$\begin{aligned} \sum_{n \in \Phi(a, s)} f_n \sum_{j=1}^t e_j z^{n+h}(P_j) &= \sum_{j=1}^t e_j z^h(P_j) \sum_{n \in \Phi(a, s)} f_n z^n(P_j) \\ &= \sum_{j=1}^t e_j z^h(P_j) f(P_j). \end{aligned} \quad (21)$$

Proposition 1: For all $A \in \mathbb{Z}_0$, the set $V(u, A) \subset K[\mathcal{X}]$ is a polynomial ideal. \square

Proof. Suppose that f and $g \in V(u, A)$ with $s := \deg(f)$ and $t := \deg(g)$. Then we show that $f+g$ and $z^h f \in V(u, A)$. Note that, by (21),

$$\begin{aligned} d(f+g)_l &= \sum_{j=1}^t e_j (f+g)(P_j) z^{l^{(s_2+t_2)}-s-t}(P_j) \\ &= \sum_{j=1}^t e_j f(P_j) z^{l^{(s_2+t_2)}-s-t}(P_j) \\ &\quad + \sum_{j=1}^t e_j g(P_j) z^{l^{(s_2+t_2)}-s-t}(P_j), \end{aligned}$$

and the last two sums are zero from the assumption and $\{l^{(s_2+t_2)} - s - t\} = \Phi(a, A - o(s) - o(t)) \subset \Phi(a, A - o(s))$, $\Phi(a, A - o(t))$ by Lemma 1. For $z^h f$, note that

$$\begin{aligned} d(z^h f)_l &= \sum_{j=1}^t e_j (z^h f)(P_j) z^{l^{(s_2+h_2)}-s-h}(P_j) \\ &= \sum_{j=1}^t e_j f(P_j) z^{l^{(s_2+h_2)}-s}(P_j), \end{aligned}$$

and $\{l^{(s_2+h_2)} - s\} = \Phi(a, A - o(s) - o(h)) + h$ by Lemma 1. Although $\Phi(a, A - o(s) - o(h)) + h \not\subset \Phi(a, A - o(s))$ in

general, the monomial $z^{l^{(s_2+h_2)}-s}$ is represented as the linear combination of elements in $\{z^l \mid l \in \Phi(a, A - o(s))\}$. Then we obtain $d(z^h f)_l = 0$ from the assumption, which completes the proof. \square

APPENDIX B

PROOF OF $V(u, B) = I(\mathcal{E})$

This follows from the next Corollary and Lemma 2.

Proposition 2: Let $f \in K[\mathcal{X}]$ be satisfying

$$\sum_{h \in \Phi(a, s)} f_h u_{h+l_j} = 0 \text{ for } l_j \in \Phi(a) \text{ with } j = 1, \dots, t$$

and $\det([z^{l_j}(P_{j'})]) \neq 0$. Then $f \in I(\mathcal{E})$ holds. \square

Proof. Since $\sum_{h \in \Phi(a, s)} f_h u_{h+l}$ is converted as (21). \square

Using Riemann–Roch Theorem, we see that the map

$$L((t+2g-1)P_\infty) \rightarrow \mathbb{F}_q^t \quad (f \mapsto [f(P_1), \dots, f(P_t)])$$

is surjective. Hence there are linearly independent t vectors of the form $[z^l(P_1), \dots, z^l(P_t)]$ for $l \in \Phi(a, t+2g-1)$, and we obtain the following sufficient condition for all errors.

Corollary: Let $f \in K[\mathcal{X}]$ be satisfying $\sum_{h \in \Phi(a, s)} f_h u_{h+l} = 0$ for all $l \in \Phi(a, t+2g-1)$. Then $f \in I(\mathcal{E})$ holds. \square

Lemma 2: We can choose a Gröbner basis $\{f^{(i)}\}_{0 \leq i < a}$ of $I(\mathcal{E})$ as $o(f^{(i)}) \leq t+2g-1+a$ for all i . \square

Proof. First, we notice that an element $f^{(i)}$ of Gröbner basis may be determined uniquely by

$$o(f^{(i)}) = \min_{f \in I(\mathcal{E})} \{o(f) \mid o(f) \equiv i \pmod{a}\}. \quad (22)$$

Let n_i be one of $\{t+2g, t+2g+1, \dots, t+2g-1+a\}$ satisfying $n_i \equiv i \pmod{a}$. We temporarily denote as $\ell(D) := \dim L(D)$, where $L(D) := \{f \in K[\mathcal{X}] \mid \text{divisor}(f) + D \text{ is positive}\} \cup \{0\}$ for a divisor D . Since we have

$$\begin{aligned} \ell((t+2g-1)P_\infty - E) &= g, \\ \ell((t+2g)P_\infty - E) &= g+1, \end{aligned}$$

\vdots

$$\ell((t+2g-1+a)P_\infty - E) = g+a,$$

where $E := \sum_{j=1}^t P_j$, there is $f \in I(\mathcal{E})$ satisfying $o(f) = n_i$. Then $o(f^{(i)}) \leq n_i$ is obtained by (22), and $\max\{o(f^{(i)}) \mid 0 \leq i < a\} \leq \max\{n_i \mid 0 \leq i < a\} = t+2g-1+a$ leads Lemma 2. \square

Proposition 3: $B \geq 2t+4g-2+a \Rightarrow V(u, B) = I(\mathcal{E})$ \square

Proof. If $f \in K[\mathcal{X}]$ and $s := \deg(f) \leq l^{(s_2)}$, then df_l is converted similarly as (21) to

$$df_l := \sum e_i f(P_i) z^{l^{(s_2)}-s}(P_i).$$

Hence, if $f(P_1) = \dots = f(P_t) = 0$, then we have $df_l = 0$, and thus $I(\mathcal{E}) \subset V(u, B)$ is obvious. To prove \supset , let $\{f_{B+1}^{(i)}\}_{0 \leq i < a}$ be a Gröbner basis of $V(u, B)$, where “ $B+1$ ” is for consistency in the previous notation. Since $I(\mathcal{E}) \subset V(u, B)$, we can choose it as $o(f_{B+1}^{(i)}) \leq t+2g-1+a$ from Lemma 2 and its proof. Now we suppose that

$d(f_{B+1}^{(i)})_l = \sum_h f_{B+1,h}^{(i)} u_{h+l(i)-s_{B+1}^{(i)}} = 0$ for all $l \in \Phi(a, B)$ with $l^{(i)} \geq s_{B+1}^{(i)}$. Then we have, by Lemma 1, $\{l^{(i)} - s_{B+1}^{(i)}\} = \Phi(a, B - o(s_{B+1}^{(i)})) \subset \Phi(a, t + 2g - 1)$. Thus we see that the inverse inclusion follows from Corollary of Proposition 2. \square

APPENDIX C GENERIC CASE

Let $m_t := \min \{m \in \mathbb{Z}_0 \mid \dim L(mP_\infty) = t\}$; recall that $\dim L(mP_\infty)$ is equal to the number of $l \in \Phi(a, m)$. If $t > g$, then we have $m_t = t + g - 1$ since $\dim L((t + g - 1)P_\infty) = t$ and $\dim L((t + g - 2)P_\infty) = t - 1$. However, for $t \leq g$, we have for example $m_6 = 10 < t + g - 1$ for Hermitian curve $y^4 + y = x^5$ over \mathbb{F}_{2^4} . We define that t -error position \mathcal{E} is *generic* if $\det([z^{l_j}(P_{j'})]) \neq 0$ for $P_{j'} \in \mathcal{E}$ and $l_j \in \Phi(a, m_t)$. If \mathcal{E} is generic, we obtain a Gröbner basis $\{f^{(i)} = z^{s^{(i)}} - \sum_{l_j \in \Phi(a, m_t)} f_{l_j}^{(i)} z^{l_j}\}$ of $I(\mathcal{E})$ by solving

$$\begin{bmatrix} z^{l_1}(P_1) & \cdots & z^{l_t}(P_1) \\ \vdots & & \vdots \\ z^{l_1}(P_t) & \cdots & z^{l_t}(P_t) \end{bmatrix} \begin{bmatrix} f_{l_1}^{(i)} \\ \vdots \\ f_{l_t}^{(i)} \end{bmatrix} = \begin{bmatrix} z^{s^{(i)}}(P_1) \\ \vdots \\ z^{s^{(i)}}(P_t) \end{bmatrix}$$

with $s^{(i)} \in \Phi(a, m_{t+i+1}) \setminus \Phi(a, m_{t+i})$. Then Lemma 2 is improved to $o(f^{(i)}) \leq t + g - 1 + a$ for generic \mathcal{E} .

Conversely, if $\det([z^{l_j}(P_{j'})]) = 0$, then the equation from the linear dependency gives $f \in I(\mathcal{E})$ with $\deg(f) \in \Phi(a, m_t)$. Thus we see that \mathcal{E} is generic if and only if the delta set $\{l \in \Phi(a) \mid l \leq s^{(l_2)}\}$ (footprint in [12]) agrees with $\Phi(a, m_t)$. Namely, our definition of generic is equivalent to the definition of generic in [23] and that of “independent” in [5].

Proposition 4: Suppose that \mathcal{E} is generic.

If $f \in V(u, m_t + o(f))$, then we have $f \in I(\mathcal{E})$. In particular, $V(u, m + a - 1) = I(\mathcal{E})$ with $m = 2t + 2g - 1$. \square

Proof. Since $\{l^{(s_2)} - s \mid l \in \Phi(a, m_t + o(f)), l^{(s_2)} \geq s\}$ agrees with $\Phi(a, m_t)$ by Lemma 1, it follows from Proposition 2. \square

APPENDIX D PROOF OF THEOREM 1

Theorem 1 is proved by the following three lemmas.

Lemma 3: Suppose that $G(z) \in V(u, M - 1)$, $dG_k \neq 0$, and $t \leq k$ with $t = \deg(G)$, $k \in \Phi^{(t_2)}(a, M)$, and $o(k) = M$. Moreover, suppose that $F(z) \in V(u, M)$ and $dF_s \neq 0$ with $s = \deg(F)$. Then, at least one condition of $s_1 \geq k_1 - t_1 + 1$ and $s_2 \neq k_2 - t_2$ holds. \square

Proof. We suppose that $s_1 \leq k_1 - t_1$ and $s_2 = k_2 - t_2$. Since $G \in V(u, M - 1)$ and $F \in V(u, M)$, we have

$$\begin{aligned} - \sum_{n \in \Phi(a, t) \setminus \{t\}} G_n u_{n+l-t} &= G_t u_l \text{ for } l \in \Phi^{(t_2)}(a, M - 1), t \leq l, \\ - \sum_{r \in \Phi(a, s) \setminus \{s\}} F_r u_{r+l-s} &= F_s u_l \text{ for } l \in \Phi^{(s_2)}(a, M), s \leq l. \end{aligned}$$

Since $n_2 + k_2 - t_2 \leq a - 1 + s_2$ and $n + k - t \geq n + s \geq s$ for $n \in \Phi(a, t)$, we have $n + k - t \in \Phi^{(s_2)}(a, M)$ and $s \leq n + k - t$

for $n \in \Phi(a, t)$, and moreover,

$$\begin{aligned} & - \sum_{n \in \Phi(a, t) \setminus \{t\}} G_n u_{n+k-t} \\ &= \sum_{n \in \Phi(a, t) \setminus \{t\}} G_n \left\{ \frac{1}{F_s} \sum_{r \in \Phi(a, s) \setminus \{s\}} F_r u_{r+(n+k-t)-s} \right\} \\ &= \frac{1}{F_s} \sum_{r \in \Phi(a, s) \setminus \{s\}} F_r \sum_{n \in \Phi(a, t) \setminus \{t\}} G_n u_{n+(r+k-s)-t} \\ &= -\frac{G_t}{F_s} \sum_{r \in \Phi(a, s) \setminus \{s\}} F_r u_{r+k-s}, \end{aligned}$$

where the last equality follows from $r + k - s \in \Phi^{(t_2)}(a, M - 1)$ and $t \leq r + k - s$ for $r \in \Phi(a, s) \setminus \{s\}$ since $r_2 + k_2 - s_2 \leq a - 1 + t_2$ and $r + k - s \geq r + t \geq t$ for $r \in \Phi(a, s)$, and the last sum agrees with $G_t u_k$ since $s_2 \leq k_2 = s_2 + t_2 \leq s_2 + a - 1$ and $k \in \Phi^{(s_2)}(a, M)$. This contradicts $dG_k \neq 0$. \square

Lemma 4: We have $s_{N,1}^{(i)} = c_{N,1}^{(i)} + 1$. \square

Proof. We prove it by induction. The case of $N = 0$ follows from the initializing. Assuming $s_{N,1}^{(i)} = c_{N,1}^{(i)} + 1$ for all i , we prove $s_{N+1,1}^{(i)} = c_{N+1,1}^{(i)} + 1$. We may assume that there is $l^{(i)} = l^{(\bar{r})}$. It follows that $s_{N,1}^{(i)} \geq l_1^{(i)} - c_{N,1}^{(\bar{r})} \Leftrightarrow s_{N,1}^{(\bar{r})} \geq l_1^{(\bar{r})} - c_{N,1}^{(i)}$. Thus we may assume that $s_{N,1}^{(i)} < l_1^{(i)} - c_{N,1}^{(\bar{r})}$, $s_{N,1}^{(\bar{r})} < l_1^{(\bar{r})} - c_{N,1}^{(i)}$, and $d_N^{(i)} \neq 0$ without loss of generality. If $d_N^{(\bar{r})} = 0$, then it contradicts Lemma 3 since $F_N^{(i)} \in V(u, N - 1)$, $F_N^{(\bar{r})} \in V(u, N)$, $s_{N,1}^{(\bar{r})} \leq l_1^{(i)} - s_{N,1}^{(i)}$, and $\bar{r} = l_2^{(i)} - i$. Thus, we obtain $d_N^{(\bar{r})} \neq 0$ and $s_{N+1,1}^{(i)} - c_{N+1,1}^{(i)} = s_{N,1}^{(\bar{r})} - c_{N,1}^{(\bar{r})}$. \square

Lemma 5: Let $F(z) \in V(u, N - 1)$, $s \leq l$ with $s = \deg(F)$ for $l \in \Phi^{(s_2)}(a, B)$, and let $G(z) \in V(u, M - 1)$, $t \leq k$ with $t = \deg(G)$ for $k \in \Phi^{(t_2)}(a, B)$. Suppose that $dG_k \neq 0$, $M = o(k) < N = o(l)$ and $k_2 - t_2 = l_2 - s_2$. Then we have

$$H(z) := dG_k z^{r-s} F - dF_l z^{r-l+k-t} G \in V(u, N),$$

and $\deg(H) = r$, where $r := s$ if $dF_l = 0$, and $r := (\max\{s_1, l_1 - k_1 + t_1\}, s_2)$ otherwise. \square

Proof. Since $r_2 = s_2$ and

$$\begin{aligned} & o(z^{r-s} F) - o(z^{r-l+k-t} G) \\ &= r_1 a + s_2 b - (r_1 - l_1 + k_1) a - t_2 b \\ &= o(l) - o(k) > 0, \end{aligned} \tag{23}$$

we obtain $\deg(H) = r$. Next, since $F \in V(u, N - 1)$ and $G \in V(u, M - 1)$, we have

$$\begin{aligned} \sum_{n \in \Phi(a, s)} F_n u_{n+p-s} &= \begin{cases} 0 & p \in \Phi^{(s_2)}(a, N - 1), s \leq p \\ dF_l & p = l, \end{cases} \\ \sum_{n \in \Phi(a, t)} G_n u_{n+p-t} &= \begin{cases} 0 & p \in \Phi^{(t_2)}(a, M - 1), t \leq p \\ dG_k & p = k. \end{cases} \end{aligned}$$

We may assume that $dF_l \neq 0$. If $p \in \Phi^{(s_2)}(a, N - 1)$ and $r \leq p$, then we have $p - l + k \in \Phi^{(t_2)}(a, M - 1)$ and $t \leq p - l + k$

from $l - k + t \leq r$, and moreover,

$$\begin{aligned} & \sum_{n \in \Phi(a,r)} H_n u_{n+p-r} \\ &= dG_k \sum_{n \in \Phi(a,s)} F_n u_{n+(r-s)+p-r} - dF_l \sum_{n \in \Phi(a,t)} G_n u_{n+(r-l+k-t)+p-r} \\ &= dG_k \sum_{n \in \Phi(a,s)} F_n u_{n+p-s} - dF_l \sum_{n \in \Phi(a,t)} G_n u_{n+(p-l+k)-t} \\ &= \begin{cases} 0 & p \in \Phi^{(s_2)}(a, N-1), r \leq p \\ dG_k \cdot dF_l - dF_l \cdot dG_k = 0 & p = l. \end{cases} \quad \square \end{aligned}$$

Proof of Theorem 1. If $d_N^{(i)} \neq 0$ and $G_N^{(\bar{i})} = 0$, then $s_{N+1,1}^{(i)} := l_1^{(i)} + 1$ and $F_{N+1}^{(i)} := x^{l_1^{(i)}+1} F_N^{(i)}$. Thus $d_{N+1}^{(i)} = 0$ and $\deg(F_{N+1}^{(i)}) = s_{N+1}^{(i)}$ hold. Supposing that $G_N^{(\bar{i})} \neq 0$, let $M < N$ be satisfying $G_N^{(\bar{i})} := (d_M^{(j)})^{-1} F_M^{(j)}$, $o(k^{(j)}) = M$, and $\bar{i} = k_2^{(j)} - j$, then we have $c_N^{(\bar{i})} = k^{(j)} - s_M^{(j)}$. Thus the theorem except for (13) and (14) follows from Lemma 5. We prove (14) by induction. The case of $N = 0$ in (14) holds by the definition. Supposing that the equality is true for $s_{N,1}^{(i)}$, we prove it for $s_{N+1,1}^{(i)}$. Let $\zeta_{N,1}^{(i)}$ be the minimum of $\zeta_{N,1}^{(i)}$ in (14). If (P), then $s_{N,1}^{(i)} = \zeta_{N,1}^{(i)} \leq \zeta_{N+1,1}^{(i)} \leq s_{N+1,1}^{(i)} = s_{N,1}^{(i)}$, thus $\zeta_{N+1,1}^{(i)} = s_{N+1,1}^{(i)}$ holds. If $d_N^{(i)} \neq 0$ and $s_N^{(i)} > l^{(i)} - c_N^{(\bar{i})}$, then we have $d_N^{(\bar{i})} \neq 0$ as in the proof of Lemma 4 and $\zeta_{N+1,1}^{(i)} \leq s_{N+1,1}^{(i)} = l_1^{(i)} - s_{N,1}^{(\bar{i})} + 1$, which is actually the equation $\zeta_{N+1,1}^{(i)} = s_{N+1,1}^{(i)}$ by Lemma 3 for $F_N^{(\bar{i})} \in V(u, N-1)$ and $F \in V(u, N)$ satisfying $\deg(F) = (\zeta_{N+1,1}^{(i)}, i)$. Finally, as for (13), if we suppose $s_{N,1}^{(i)} < s_{N,1}^{(j)}$ with $i < j$, then we have $y^{j-i} F_N^{(i)} \in V(u, N-1)$ and $\deg(y^{j-i} F_N^{(i)}) = (s_{N,1}^{(i)}, j)$, which contradict the minimality of $s_{N,1}^{(j)}$. \square

Thus we have proved the theorem for an algorithm that is not a parallel version, i.e., the algorithm with direct calculation of $d_N^{(i)}$ by (5) without $v_N^{(i)}$ and $w_N^{(i)}$. To prove our parallel inverse-free BMS algorithm described in Section III, we have to show further that $d_N^{(i)}$ is obtained by the coefficient of $v_N^{(i)}$; we omit this procedure and refer to similar cases [14][16] of ordinary parallel BMS algorithm.

REFERENCES

- [1] E. R. Berlekamp, *Algebraic coding theory*, McGraw-Hill, New York, 1968.
- [2] D. Cox, J. Little, D. O'Shea, *Ideals, varieties, and algorithms*, UTM Springer-Verlag, 1992.
- [3] G.-L. Feng, T. R. N. Rao, "Decoding algebraic-geometric codes up to the designed minimum distance," *IEEE Trans. Inf. Theory*, vol.39, pp. 37–45, Jan. 1993.
- [4] V. Guruswami, M. Sudan, "Improved decoding of Reed–Solomon and algebraic-geometric codes," *IEEE Trans. Inf. Theory*, vol.45, no.6, pp.1757–1767, Sep. 1999.
- [5] H. E. Jensen, R. R. Nielsen, T. Høholdt, "Performance analysis of a decoding algorithm for algebraic-geometry codes," *IEEE Trans. Inf. Theory*, vol.45, no.5, pp.1712–1717, July 1999.
- [6] J. Justesen, K. J. Larsen, H. E. Jensen, A. Havemose, T. Høholdt, "Construction and decoding of a class of algebraic geometry codes," *IEEE Trans. Inf. Theory*, vol.35, no.4, pp.811–821, July 1989.
- [7] R. Köter, "A fast parallel implementation of a Berlekamp–Massey algorithm for algebraic-geometric codes," *IEEE Trans. Inf. Theory*, vol.44, no.4, pp.1353–1368, July 1998.
- [8] R. Koetter, A. Vardy, "Algebraic soft-decision decoding of Reed–Solomon codes," *IEEE Trans. Inf. Theory*, vol.49, no.11, pp.2809–2825, Nov. 2003.
- [9] K. Lee, M. E. O'Sullivan, "An interpolation algorithm using Gröbner bases for soft-decision decoding of Reed–Solomon codes," *Proc. IEEE Int. Symp. Information Theory*, Seattle, July 2006.
- [10] K. Lee, M. E. O'Sullivan, "Sudan's list decoding of Reed–Solomon codes from a Gröbner basis perspective," arXiv:math.AC/0601022.
- [11] J. L. Massey, "Shift-register synthesis and BCH decoding," *IEEE Trans. Inf. Theory*, vol.IT-15, pp.122–127, Jan. 1969.
- [12] H. Matsui, S. Mita, "Footprint of polynomial ideal and its application to decoder for algebraic-geometric codes," *Proc. Int. Symp. Information Theory and Its Applications (ISITA)*, pp.1473–1478, Parma, Italy, Oct. 2004.
- [13] H. Matsui, E. Yamamoto, S. Mita, "Reduction of decoder for codes on algebraic curves as an application of footprint," (in Japanese) *Proc. 27th Symp. Information Theory and Its Applications (SITA)*, pp.471–474, Dec. 2004.
- [14] H. Matsui, S. Sakata, M. Kurihara, "Fast parallel decoding on systolic array architecture for codes on a class of algebraic curves," *RIMS Kokyuroku (Lecture note at Research Institute for Mathematical Sciences in Kyoto University)*, vol.1420Cp.193–205, April 2005.
- [15] H. Matsui, S. Mita, "On small-scale decoders for codes on C_a^b curves," *Proc. Hawaii, IEICE and SITA Joint Conference on Information Theory*, IT2005-16, pp.91–95, May 2005.
- [16] H. Matsui, S. Sakata, M. Kurihara, S. Mita, "Systolic array architecture implementing Berlekamp–Massey–Sakata algorithm for decoding codes on a class of algebraic curves," *IEEE Trans. Inf. Theory*, vol.51, no.11, pp.3856–3871, Nov. 2005.
- [17] H. Matsui, "On the smallest-scale decoder for codes on algebraic curves," *Proc. 28th Symp. Information Theory and Its Applications (SITA)*, pp.547–550, Nov. 2005.
- [18] H. Matsui, S. Mita, "Inverse-free implementation of Berlekamp–Massey–Sakata algorithm for decoding codes on algebraic curves," *Proc. IEEE Int. Symp. Information Theory*, pp.2250–2254, Seattle, July 2006.
- [19] H. Matsui, S. Mita, "Efficient encoding via Gröbner bases and discrete Fourier transforms for several kinds of algebraic codes," Submitted in 2007 *IEEE Int. Symp. Information Theory*, arXiv:cs.IT/0703104.
- [20] R. Matsumoto, "The C_a^b curve—A generalization of the Weierstrass form to arbitrary plane curves," in his website.
- [21] S. Miura, "Algebraic geometric codes on certain plane curves," (Japanese) *Trans. IEICE*, J75–A, no.11, pp.1735–1745, Nov. 1992.
- [22] Y. Numakami, M. Fujisawa, S. Sakata, "A fast interpolation algorithm for list decoding of Reed–Solomon codes," (Japanese) *Trans. IEICE*, vol.J83–A, no.11, pp.1309–1317, Nov. 2000.
- [23] M. E. O'Sullivan, "Decoding Hermitian codes beyond $(d_{\min} - 1)/2$," *Proc. IEEE International Symposium on Information Theory*, p.384, Ulm, Germany, June 29–July 4, 1997. See also "Decoding of Hermitian codes: Beyond the minimum distance bound," in his website.
- [24] M. E. O'Sullivan, "On Koetter's algorithm and the computation of error values," *Designs, Codes and Cryptography*, vol.31, pp.169–188, 2004.
- [25] S. Sakata, "Finding a minimal set of linear recurring relations capable of generating a given finite two dimensional array," *Journal of Symbolic Computation*, no.5, pp.321–337, Nov. 1988.
- [26] S. Sakata, H. E. Jensen, T. Høholdt, "Generalized Berlekamp–Massey decoding of algebraic geometric code up to half the Feng–Rao bound," *IEEE Trans. Inf. Theory*, vol.41, no.6, Part I, pp.1762–1768, Nov. 1995.
- [27] S. Sakata, M. Kurihara, "A systolic array architecture for fast decoding of one-point AG codes and scheduling of parallel processing on it," *Proc. AAECC-13 (Eds., M. Fossorier, H. Imai, S. Lin, A. Poli), Lecture Notes in Computer Science*, vol.1719, pp.302–313, Springer Verlag, 1999.
- [28] S. Sakata, "Applications of the BMS algorithm to decoding of algebraic codes," preprint for Workshop on Gröbner bases in Cryptography, Coding Theory and Algebraic Combinatorics, Linz, April 30–May 5, 2006.
- [29] M. Sudan, "Decoding of Reed–Solomon codes beyond the error-correction bound," *Journal of Complexity*, vol.13, no.1, pp.180–193, 1997.
- [30] Y. Sugiyama, M. Kasahara, S. Hirasawa, T. Namekawa, "A method for solving key equation for decoding Goppa codes," *Information and Control*, vol.27, pp.87–99, 1975.